TOPIC: 6.857 - Network & Computer Security

DATE: 3/14/12

FILE UNDER:

P L11.1

Admin:

Pset #3 due Monday (3/19)

Form your project groups! Choose a topic...

Today:

- Hash fn applications (continued)
- Hash fn construction (Merkle-Damgård)
- "Merkle-Damgård Revisited" (Coron, Dodis, Malinaud, Puniya)
- Floyd's "Two-Finger Algorithm" for finding collisions.

TOPIC:

DATE:

FILE UNDER:

④ Commitments

- Alice has value x (e.g. auction bid)
- Alice computes $C(x)$ ("commitment to x") & submits $C(x)$ as her "sealed bid"
- When bidding has closed, Alice should be able to "open" $C(x)$ to reveal x
- <u>Binding property</u>: Alice should not be able to open $C(x)$ in more than one way! (She is committed to just one x.)
- <u>Secrecy (hiding)</u>: Auctioneer (or anyone else) seeing $C(x)$ should not learn anything about x.
- <u>Non-malleability</u>: Given $C(x)$, it shouldn't be possible to produce $C(x+1)$, say...
- <u>How</u>:

$$C(x) = h(r \| x) \qquad r \in_R \{0,1\}^{256}$$

To open: reveal r & x

- Note that this method is <u>randomized</u> (as it must be for secrecy.
- Need: OW, CR, NM

(really need more, for secrecy, as $C(x)$ should not reveal partial information about x, even.)

TOPIC:

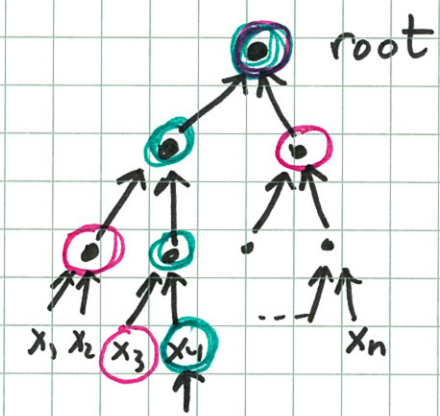
DATE:

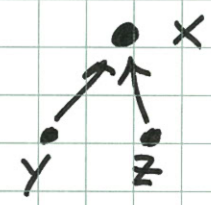
FILE UNDER:

PAGE: L11.3

⑤ <u>To authenticate a collection of n objects:</u>

Build a tree with n leaves $x_1, x_2, \ldots, x_n$ & compute authenticator node as fn of values at children... This is a "Merkle tree":





value at x

= h(value at y || value at z)

Root is authenticator for all n values $x_1, x_2, \ldots, x_n$

To authenticate x_i, give sibling of x_i & sibling of all his ancestors up to root

Apply to: time-stamping data

authenticating whole file system

<u>Need:</u> <u>CR</u>

Hash function construction ("Merkle-Damgard" style)

- Choose output size d (e.g. $d = 256$ bits)
- Choose "chaining variable" size c (e.g. $c = 512$ bits)

  [Must have $c \geq d$; better if $c \geq 2 \cdot d$ ...]
- Choose "message block size" b (e.g. $b = 512$ bits)
- Design "compression function" f

$$f : \{0,1\}^c \times \{0,1\}^b \rightarrow \{0,1\}^c$$

  [f should be OW, CR, PR, NM, TCR, ...]
- Merkle-Damgard is essentially a "mode of operation" allowing for variable-length inputs:

  * Choose a c-bit initialization vector IV, c_0

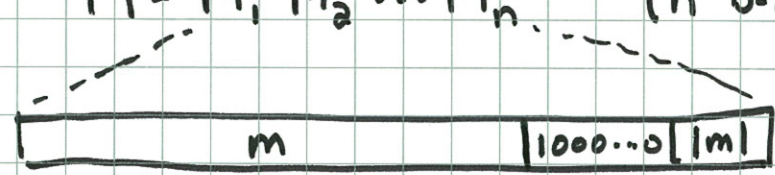
    [Note that c_0 is fixed & public.]

  * [Padding] Given message, append

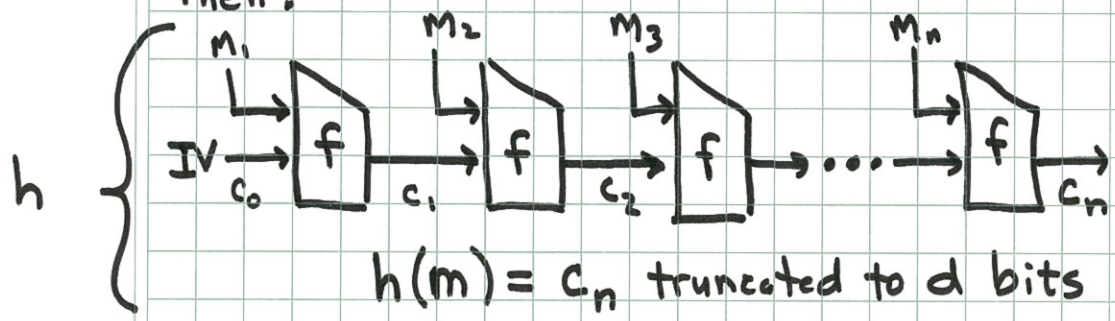
    - 10^* bits
    - fixed-length representation of length of input

  so result is a multiple of b bits in length:

$$M = M_1 M_2 \cdots M_n \qquad (n \text{ } b\text{-bit blocks})$$



Then:



$h(m) = c_n$ truncated to d bits

<u>Theorem:</u> If f is CR, then so is h.

<u>Proof:</u> Given collision for h, can find one for f by working backwards through chain. ∎

<u>Thm:</u> Similarly for OW.

<u>Common design pattern for f:</u>

$$f(C_{i-1}, M_i) = C_{i-1} \oplus E(M_i, C_{i-1})$$

where $E(K, M)$ is an encryption function (block cipher) with b-bit key and c-bit input/output blocks.

(Davies-Meyer construction)

"Merkle-Damgård Revisited" (Coron, Dodis, Malinaud, Puniya)

Is MD a "good" method?

What does this mean?

Suppose that f is a random oracle (fixed input length)

$$f : \{0,1\}^{b+c} \to \{0,1\}^{c}$$

Then is MD^f indistinguishable from a VIL RO?

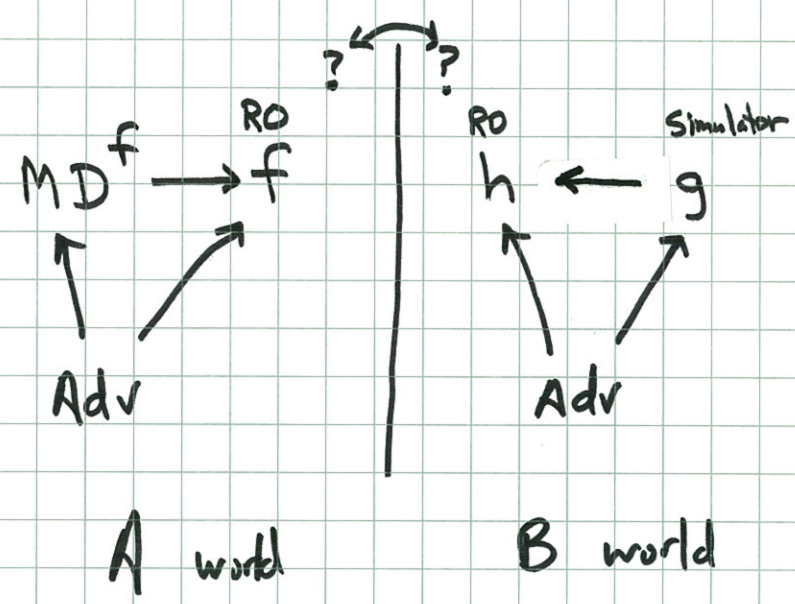
(VIL = "variable input length")

Adversary has access to:

(A) MD^f and also to f (f is FIL)

(B) RO h and also to g (h is VIL & g FIL)

where g is constructed to bear same relation to h as f does to MD^f ("simulator")



Note: g may call h, but doesn't see Adv's calls to h.

Standard construction MD^f fails (for $c=d$):

Can't build simulator g to bear right relation to h (i.e. so that h appears to be MD^g)

<u>Example of problem (message extension):</u> (sketch)

h & g should satisfy

$$MD^g(m_1 \| m_2) = h(m_1 \| m_2) = g(g(IV, m_1), m_2)$$

Adv: [computes $u = h(m_1)$
computes $v = g(u, m_2)$ (*)
computes $w = h(m_1, m_2)$
if $v = w$: answer "A world"
else: answer "B world"]

Adv always right in A world, and almost always right in B world, since simulator g doesn't know how to answer query (*). [It didn't see query for u, so even though it can access h, it doesn't have ability to figure out m_1, and so reply to (*) in way that makes it consistent with $h(m_1, m_2)$.]

But, it is not hard to fix MD construction so it becomes "indistinguishable from RO" (given FIL RO f) [technically this is called "indifferentiability"].

<u>Four methods:</u> (any work to fix MD)

① <u>Encode m to be "prefix-free"</u> before applying MD:

e.g. $0 \| m_1 \| 0 \| m_2 \| 0 \| m_3 \| \cdots \| 1 \| m_n$

<u>or</u> $L \| m_1 \| m_2 \| \cdots \| m_n$

length of message in bits
m_n padded with 10^*

② <u>Drop output bits:</u>

Let $d = c/2$. Drop $c/2$ bits of output.

③ NMAC construction

$$g(MD^f(M))$$

[g indep. function from $\{0,1\}^c$ to $\{0,1\}^d$]

④ HMAC construction:

$$MD^f(MD^f(M))$$

* With such methods, it is then "safe" to treat (modified) MD^f as a RO (assuming f is indistinguishable from a FIL R.O.)

Floyd's "Two-Finger" algorithm for finding collisions

Let $f : \{0,1\}^n \to \{0,1\}^n$

Pick random x_0.

Let $x_{i+1} = f(x_i)$ for $i = 1, 2, \ldots,$

eventually loops:

$$\underbrace{x_0 \to x_1 \to x_2 \to \cdots \to x_{t-1}}_{\text{tail of length } t} \to \underbrace{x_t \to x_{t+1} \cdots\cdots x_d \to \cdots x_{t+c-1}}_{\text{cycle of length } c}$$

If f is "random":

$$E(t) = E(c) = \Theta(\sqrt{2^n}) = \Theta(2^{n/2}) \quad \text{B.P.}$$

Two Finger alg:

Finger 1: $x_0, x_1, x_2, \ldots$ (single speed)

Finger 2: $x_0, x_2, x_4, \ldots$ (double speed)

until $x_d = x_{2d}$ [Lemma: $d \equiv 0 \pmod{c}$]

Then:

Finger 1 starts at $x_d, x_{d+1}, \ldots$

Finger 2 " " $x_0, x_1, \ldots$

} both single speed

until they collide at x_t

Previous step gives $f(x_{t-1}) = f(x_{t+c-1})$ collision!